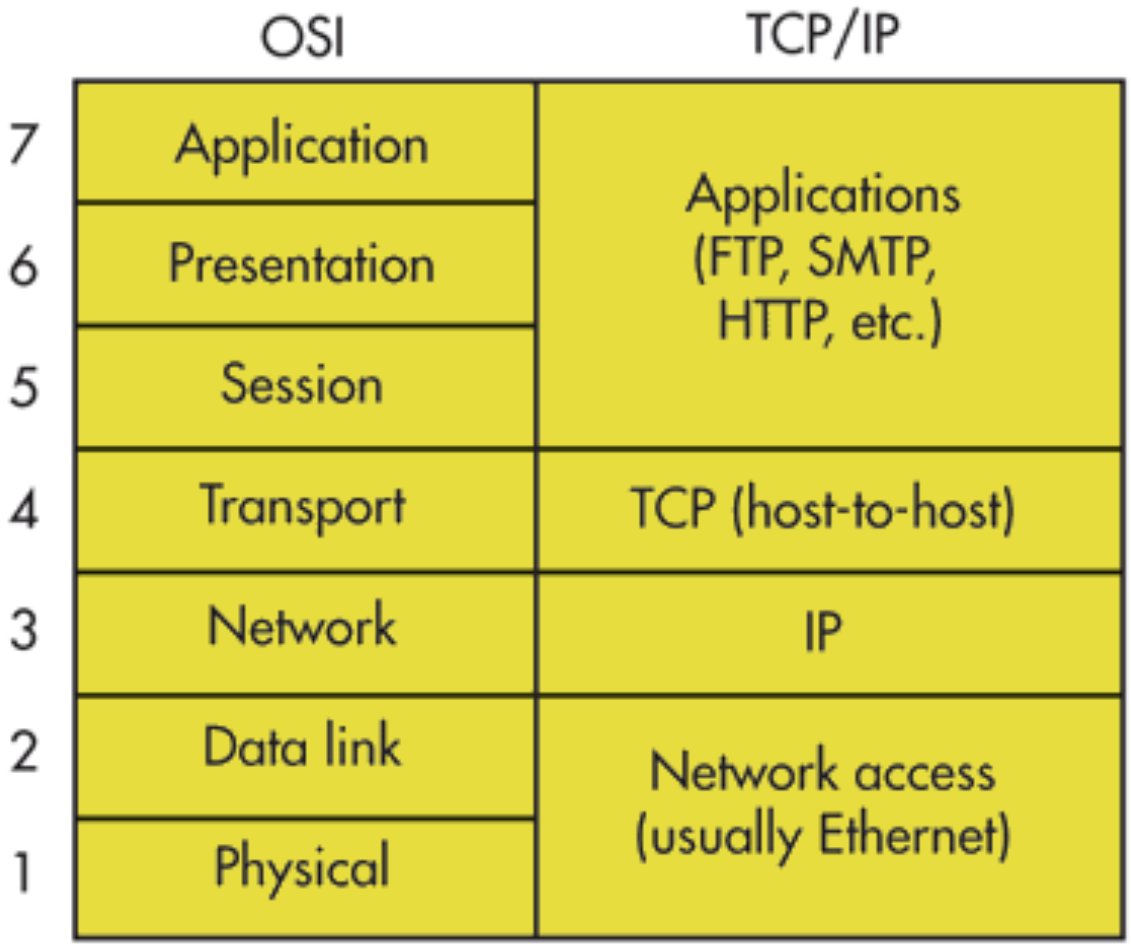
Môn: Mạng máy tính

LẬP TRÌNH SOCKET

Trình bày:
GVHDTH: Chung Thùy Linh
ctlinh@fit.hcmus.edu.vn

OSI & TCP/IP MODEL



Socket: IP, port, TCI

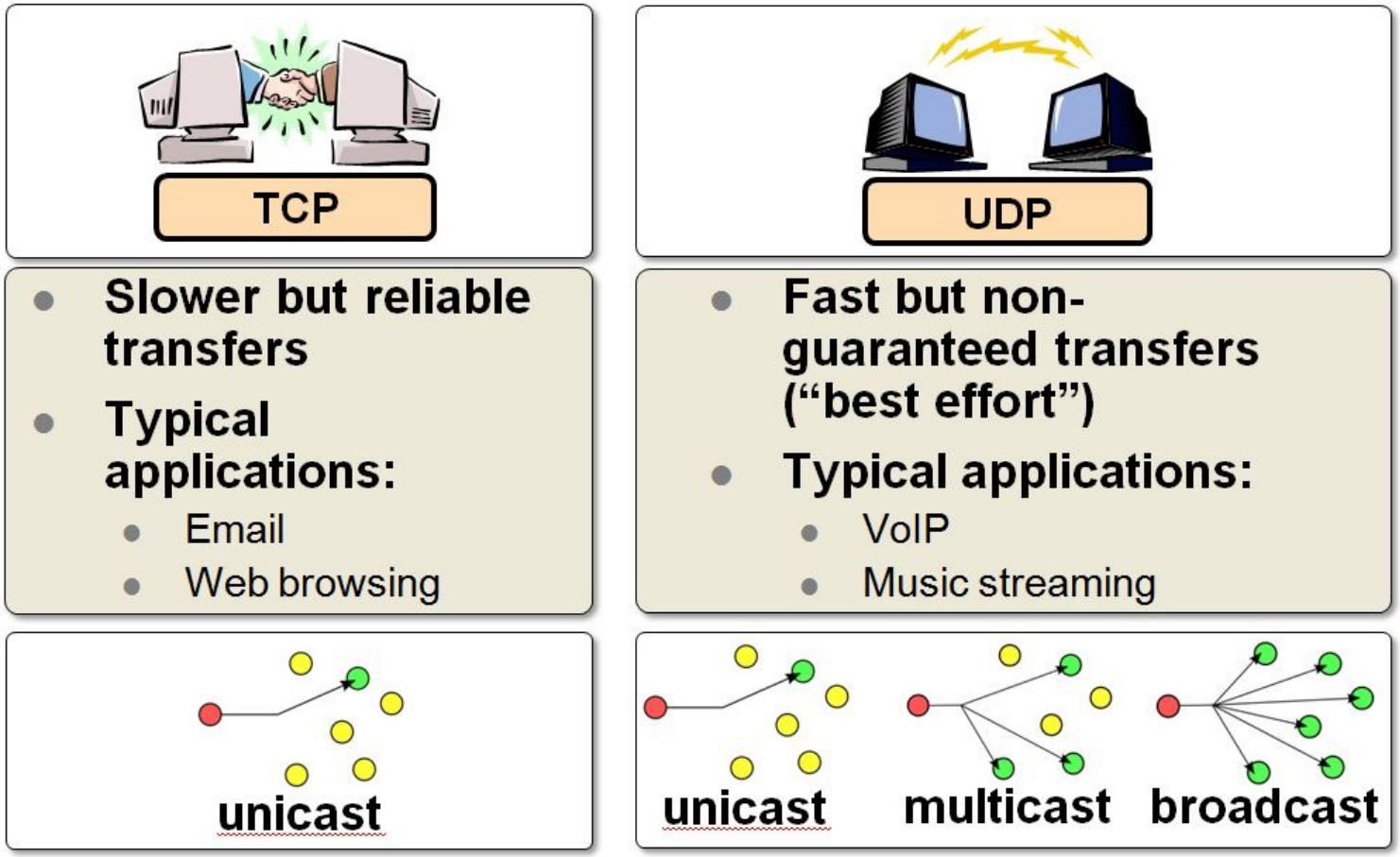
Source IP:
Destination IP:

Source MAC:
Destination MAC:

This Photo by Unknown Author is licensed under CC BY-NC-ND

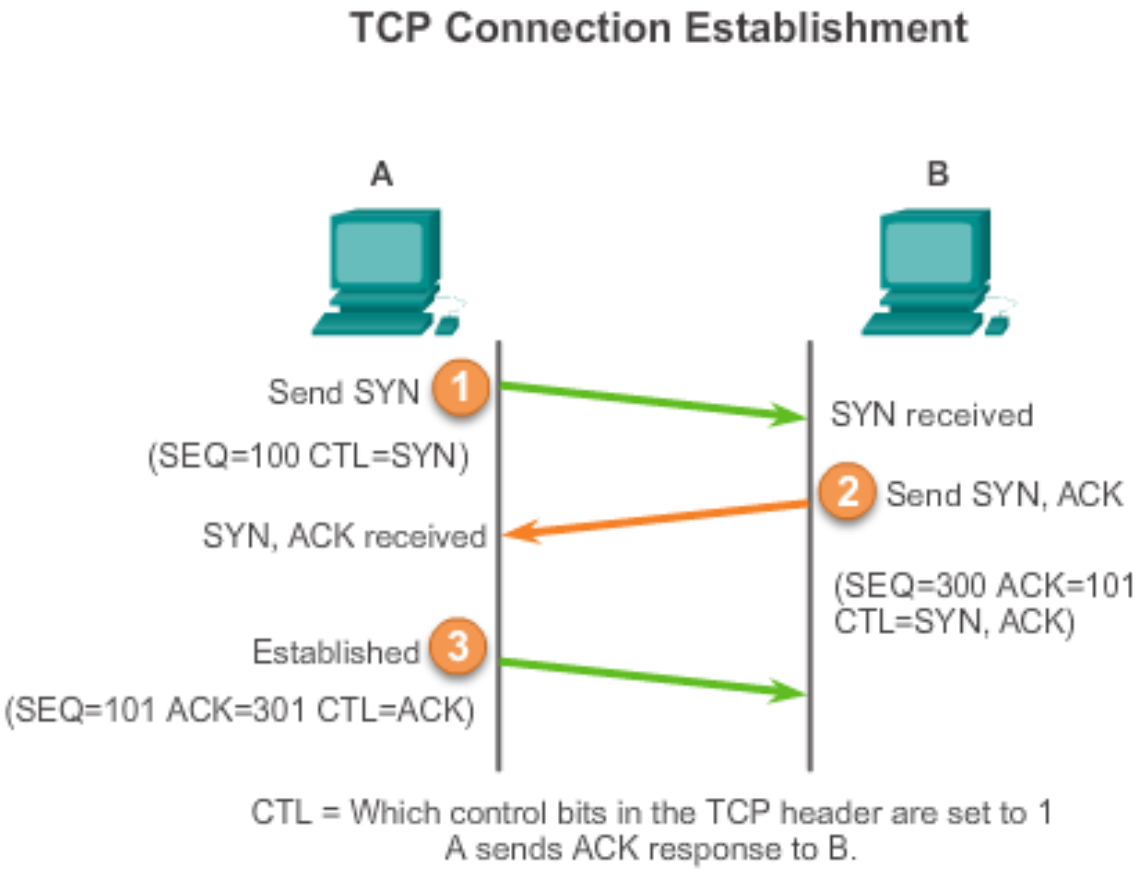
TCP vs UDP

3-ways handshake



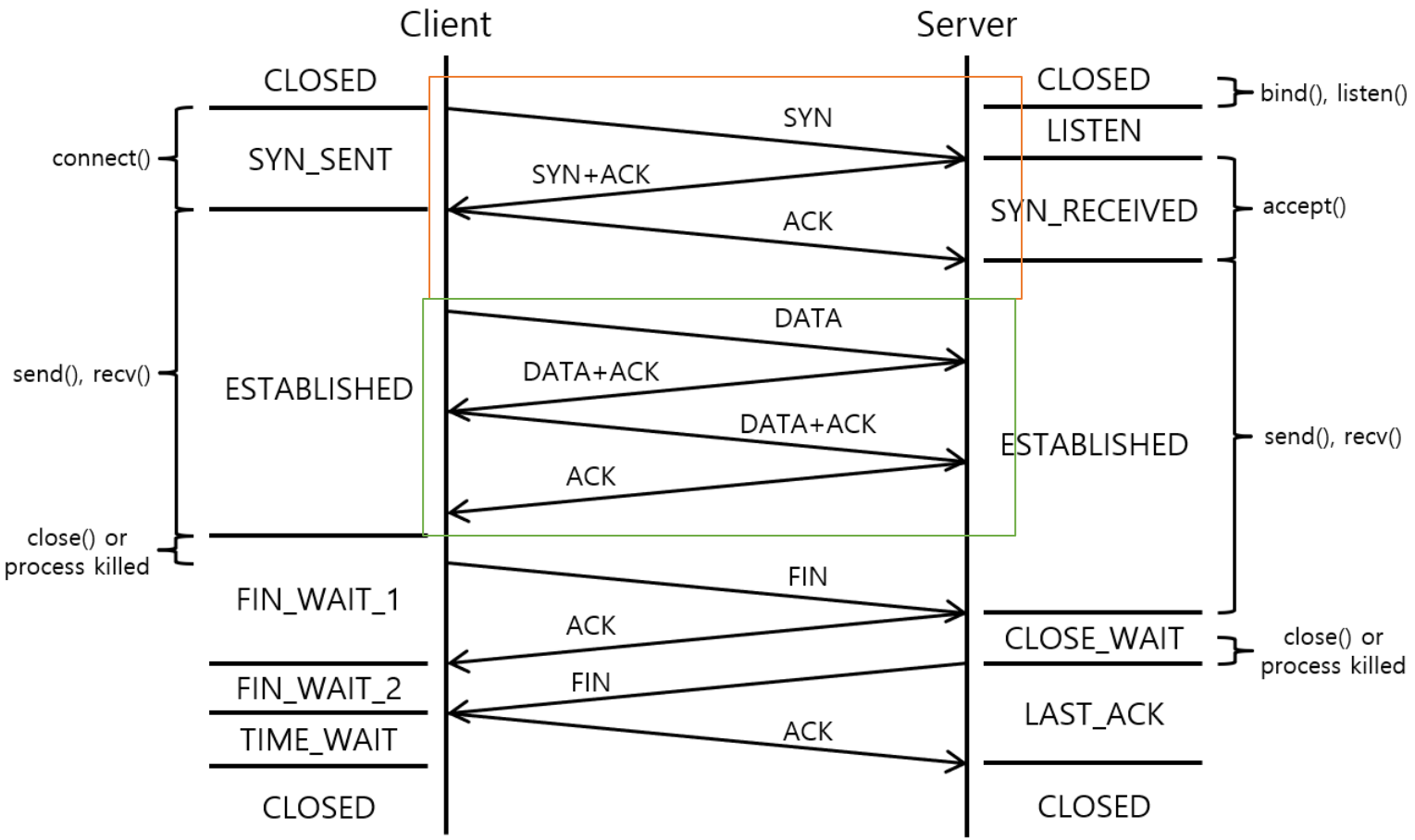
This Photo by Unknown Author is licensed under CC BY-SA-NC

TCP



This Photo by Unknown Author is licensed under CC BY-SA

TCP



This Photo by Unknown Author is licensed under CC BY-SA-NC

Nội dung chính

1. Giới thiệu Socket
2. Hướng dẫn viết 1 ứng dụng S-C
3. Một số hàm cơ bản trong CSocket
4. Demo

Giới thiệu về Socket

- Sockets cung cấp một interface để lập trình mạng tại tầng Transport.
- Một socket là một end-point của một liên kết giữa hai ứng dụng mạng.
- Nhiều NNLT: C, C++, Java, VB, C#, . . .
- Windows Socket Application Programming Interface (Winsock API)
- Winsock hỗ trợ xây dựng các ứng dụng mạng trên nền TCP/IP.

Nội dung chính

1. Giới thiệu Socket
2. Hướng dẫn viết 1 ứng dụng S-C
3. Một số hàm cơ bản trong CSocket
4. Demo

Xây dựng UD Client-Server với Socket

Các bước cần thực hiện:

1. Xác định kiến trúc mạng: Client – Server, Peer-to-Peer
2. Giao thức sử dụng tầng Transport: TCP, UDP
3. Các port sử dụng ở Server và Client :

- DHCP: cung cấp IP động cho client. Port server: 67, port client 68
- HTTP: port server 80, port client (>1024)
- HTTPs: port server 443, port client (>1024)
- Câu hỏi tìm hiểu: range port có thể sử dùng cho ứng dung mạng?

4. Giao thức tầng ứng dụng khi trao đổi dữ liệu giữa hai end-host

- Thứ tự gửi thông điệp
- Nội dung thông điệp
- Cấu trúc thông điệp (string, file, int, float, struct,…)

5. Lập trình

Xây dựng UD Client-Server với Socket

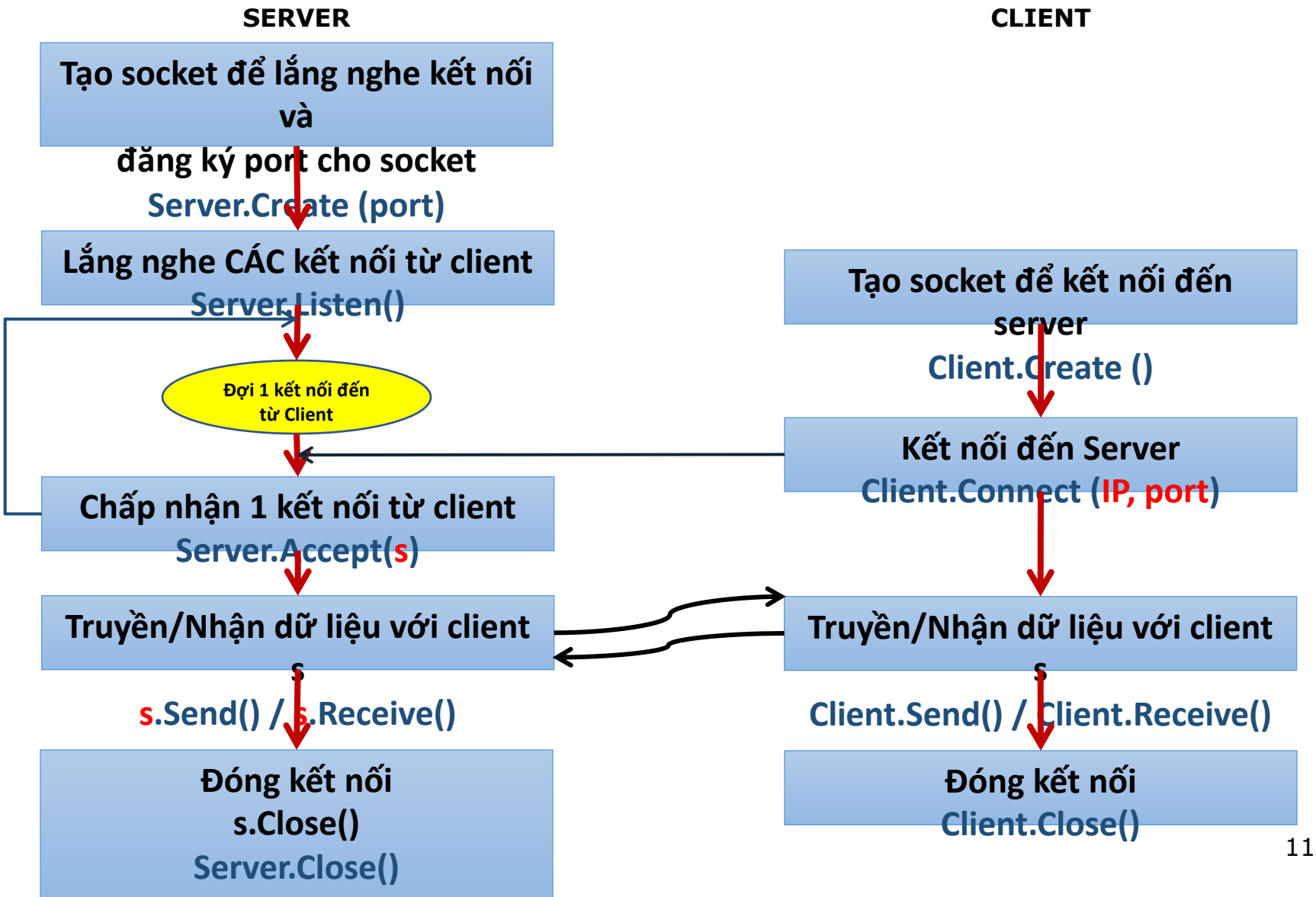
1/ Xác định bài tập và đồ án:

- Kiến trúc Client – Server
- Giao thức sử dụng tầng Transport là TCP

2/ Ghi chú:

- XD UD theo giao thức tầng UD được định nghĩa sẵn → phải tuân thủ đúng quy định → tham khảo RFC (request for comment)
- Peer-to-Peer: 1 UD có cả Client - Server

Xây dựng UD Client-Server với Socket



Nội dung chính

1. Giới thiệu Socket
2. Hướng dẫn viết 1 ứng dụng S-C
3. Một số hàm cơ bản trong CSocket
4. Demo

Một số hàm cơ bản trong lớp CSocket

- CSocket (được hỗ trợ trong MFC) quản lý việc truyền và nhận dữ liệu thông qua socket
- Tham khảo trong MSDN: http://msdn.microsoft.com/en-US/library/65bbyctt%28v=VS.80%29.aspx

Một số hàm cơ bản trong lớp CSocket

Khởi tạo Socket

```
BOOL AfxSocketInit(
WSADATA* lpwsaData = NULL
);
```

- Trước khi sử dụng các hàm của lớp CSocket, chúng ta phải gọi hàm này để khởi tạo Windows Socket với tham số lpwsaData gán bằng NULL.

Một số hàm cơ bản trong lớp CSocket

Tạo socket

```
BOOL Create(
UINT nSocketPort = 0,
int nSocketType = SOCK_STREAM,
LPCTSTR lpszSocketAddress = NULL
);
```

- nSocketPort: port của socket; nêu băng 0 thì port sẽ được MFC chọn ngẫu nhiên.
- nSocketType là SOCK_STREAM (TCP) hay SOCK_DGRAM (UDP).
- lpszSocketAddress: địa chỉ IP của host dùng socket. Nếu dùng NULL, socket sẽ lắng nghe hoạt động của client trên tất cả các card mạng.
- Giá trị trả về: khác 0 nếu thành công; ngược lại thì bằng 0 và mã lỗi cụ thể sẽ được cho khi gọi hàm GetLastError.

Một số hàm cơ bản trong lớp CSocket

Lắng nghe các yêu cầu kết nối:

```
BOOL Listen(
int nConnectionBacklog = 5
);
```

- Hàm này chỉ hỗ trợ cho socket dạng SOCK_STREAM.
- nConnectionBacklog: chiều dài tối đa mà hàng đợi kết nối chưa được chấp nhận có thể tăng.
- Giá trị trả về: khác 0 → thành công, bằng 0 → thất bại.

Một số hàm cơ bản trong lớp CSocket

Chấp nhận một kết nối

```
virtual BOOL Accept(
CAsyncSocket& rConnectedSocket,
SOCKADDR* lpSockAddr = NULL,
int* lpSockAddrLen = NULL );
```

- rConnectedSocket: tham chiếu định danh socket của kết nối được chấp nhận.
- lpSockAddr trỏ đến cấu trúc SOCKADDR nhận địa chỉ IP của socket kết nối đến.
- Nếu lpSockAddr hay lpSockAddrLen lấy giá trị mặc định NULL thì sẽ không có thông tin từ socket (trên client) được chấp nhận được trả về.
- lpSockAddrlen chứa chiều dài thực sự của lpSockAddr khi trả về theo byte.
- Giá trị trả về: khác 0 nếu thành công và bằng 0 nếu thất bại.

Một số hàm cơ bản trong lớp CSocket

Kết nối đến server

```
BOOL Connect(
LPCTSTR lpszHostAddress,
UINT nHostPort
);
```

- lpszHostAddress: địa chỉ IP của Server.
- nHostPort: Port của socket lắng nghe kết nối trên Server.
- Giá trị trả về: khác 0 nếu thành công; ngược lại, thất bại = 0.

Một số hàm cơ bản trong lớp CSocket

Truyền/nhận dữ liệu

```
virtual int Send/Receive(
const void* lpBuf,
int nBufLen,
int nFlags = 0
);
```

- lpBuf: vùng đệm chứa dữ liệu.
- nBuffLen: kích thước của vùng đệm tính theo byte.
- nFlag: cách nhận/truyền dữ liệu, sử dụng giá trị mặc định là 0.
- Giá trị trả về là số byte nhận/truyền được

Thí dụ minh họa

Bài toán 1: viết một ứng dụng chat tuần tự giữa Server – Client (Server chat trước).

Xác định 4 bước trước khi lập trình:

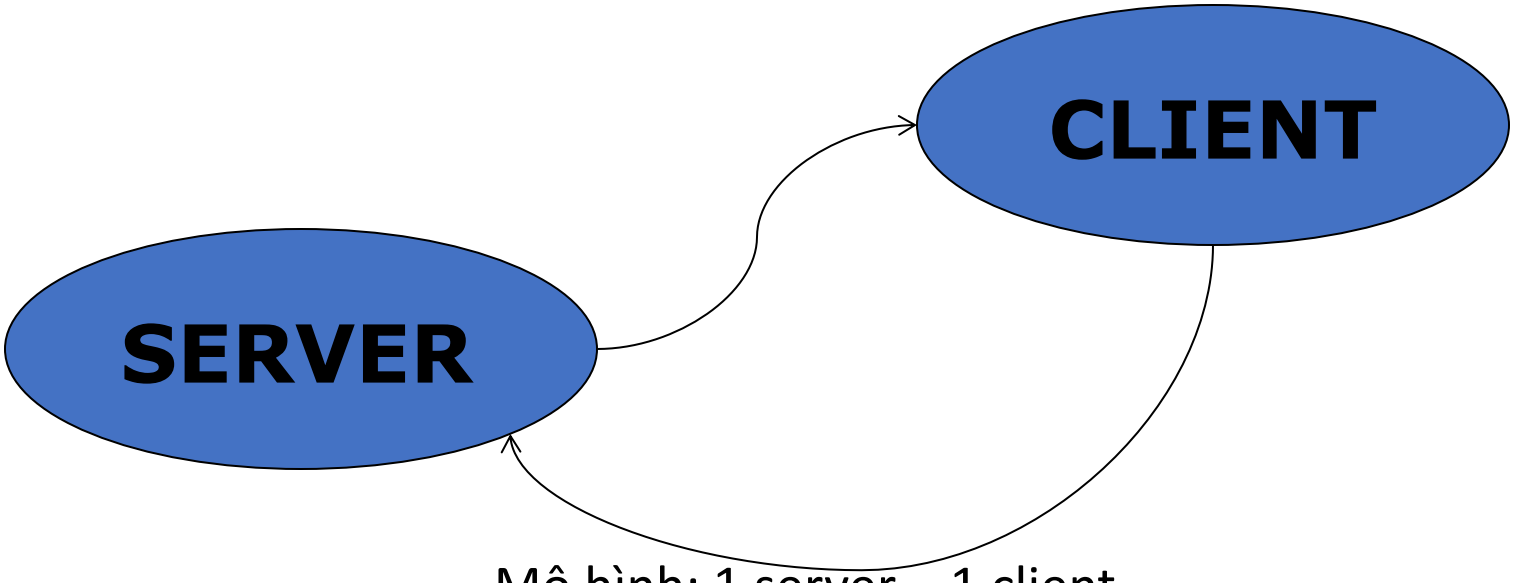
1. Xác định kiến trúc mạng: ?
2. Giao thức sử dụng tầng Transport: ?
3. Các port sử dụng ở Server và Client: Server - ?
4. Giao thức tầng ứng dụng khi trao đổi dữ liệu giữa hai end-host:

   + server → client → server → client → …

   + format thông điệp truyền giữa client và server: <Chiều dài thông điệp><Thông Điệp>

Thí dụ minh họa

Xét kịch bản trong bài toán 1



Mô hình: 1 server – 1 client

Thí dụ minh họa

Bài toán 2: N clients sẽ gửi cho server 2 số từ 0 → 10. Các client luân phiên nhau gửi theo 2 lượt. Server đếm số lượng các số Nguyên tố được gửi lên và thông báo kết quả cho tất cả client.

Xác định 4 bước trước khi lập trình: ?

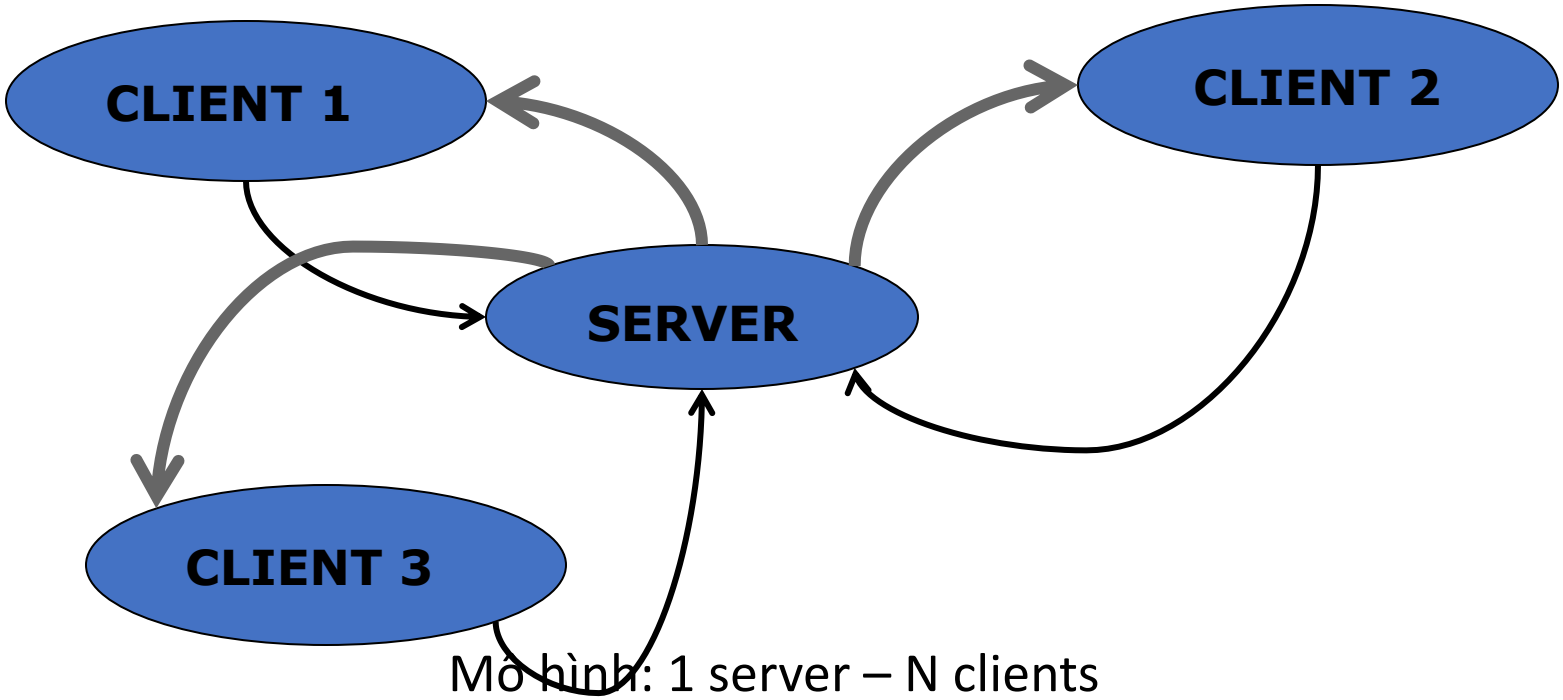
- Giao thức tầng ứng dụng khi trao đổi dữ liệu giữa hai end-host:

  + N clients → server → N clients …

  + format thông điệp truyền giữa client và server: <Kích thước số nguyên><Số nguyên>

Thí dụ minh họa

Xét kịch bản trong bài toán 2



Mô hình: 1 server – N clients

Nội dung chính

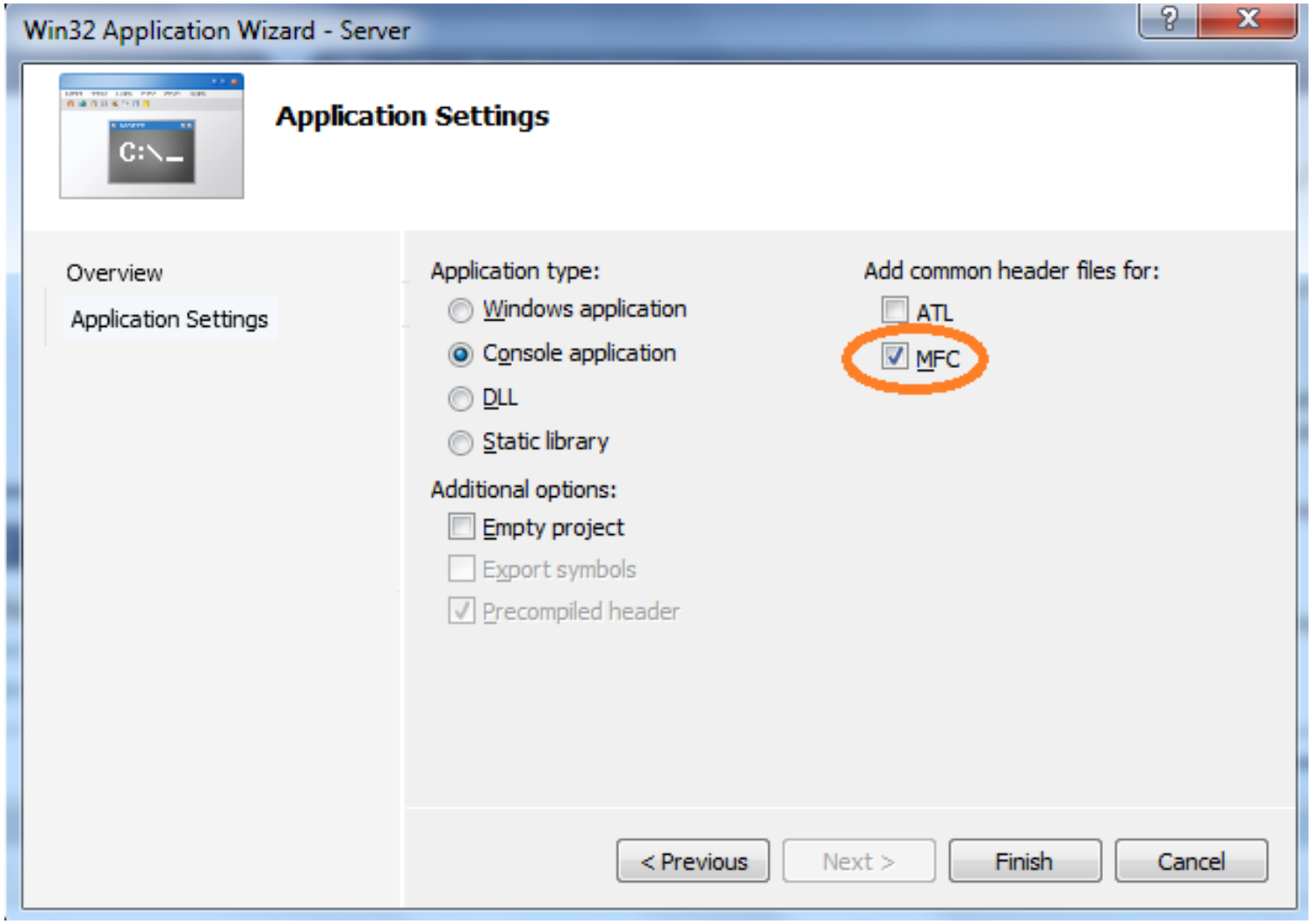
1. Giới thiệu Socket
2. Hướng dẫn viết 1 ứng dụng S-C
3. Một số hàm cơ bản trong CSocket
4. Demo

Demo

- Cách tạo 1 project
- Code demo Bài toán 1: Server và Client chat tuần tự
- Code demo Bài toán 2: N clients gửi lên Server, Server tính toán gửi trả kết quả về cho các clients
- Giải thích 1 số kịch bản trong đề
  - Xét kịch bản đăng ký tên (nickname)
  - Xét kịch bản server phát sinh “lá bài” gửi cho clients

Demo: cách tạo 1 project

Demo: cách tạo 1 project



Demo code chat 1S – 1C

Server (include "afxsock.h")

```
CSocket server, s; //khai báo biến

AfxSocketInit(NULL);  // kt Windows Socket

if (!server.Create(1234))  { // có port
        printf("That bai roi nhe");
        exit(); } // tạo socket
server.Listen();  // lắng nghe
server.Accept(client) ; // chấp nhận kết nối
do  {
        printf("\nServer: ");
        gets(s_str);
        len = strlen(s_str);
        s.Send(s_str,len,0);
        len = s.Receive(r_str,100,0);
        // gán ký tự kết thúc chuỗi
        r_str[len] = 0;
        // hiển thị chuỗi nhận được
        printf("\nCleint: %s",r_str);
}while(strcmp(r_str,"exit")&&strcmp(s_str,"exit"));
s.Close();
server.Close();
```

Client (include "afxsock.h")

```
CSocket client;

AfxSocketInit(NULL);

client.Create();  // không có port


client.Connect(svrAddr,1234) ; // kết nối

do {
        len = client.Receive(r_str,100,0);
        r_str[len] = 0;
        printf("\n Server: %s",r_str);
        printf("\n Client: ");
        gets(s_str);
        client.Send(s_str,strlen(s_str),0);
}while(strcmp(r_str,"exit")&&strcmp(s_str,"exit"));



client.Close();
```

Thanks for your attention !